Интерфейси

Кристиян Стоименов

25 март 2024 г.

ТУЕС,
ПВМКС

Какво е интерфейс? i

- Неформално, под интерфейс ще разбираме установен начин за комуникация между отделни подсистеми.
- Някои основни характеристики, от които ще се интересуваме включват дали интерфейсът е синхронен, или асинхронен; колко участници в комуникацията могат да изпращат съобщения едновременно; кои участници могат да инициират комуникация; дали съобщенията се изпращат в един „сноп", или в много едновременно; и разбира се скорост на предаване на информация.
- Някои популярни примери са USB и PCIe.
- Ние ще се запознаем по-подробно с UART, I2C и SPI.

Какво е интерфейс? ii

- За да разглеждаме интерфейси отблизо, обичайно се използва осцилоскоп и/или логически анализатор.

UART i

- Universal Asynchronous Receiver-Transmitter.
- Асинхронен, full-duplex, сериен (последователен).
- Използват се два проводника, всеки от които има *tx* и *rx* страни - от едната се изпраща към втората. Понеже UART е full-duplex, това означава, че всяко устройство, което участва в такава комункация може както да изпраща данни, така и да получава едновременно.

UART ii

- Тъй като UART е асинхронен, това означава, че няма допълнителен сигнал, който да предава тактов сигнал. Поради тази причина, за да бъде установена успешна комуникация, то устройствата, участващи в нея трябва да имат предварително определена скорост (напр. 9600 или 115200).
- Последователността на интерфейса пък означава, че във всяко съобщение данните се изпращат бит по бит (за разлика от паралелни интерфейси, при които се изпращат повече на брой битове едновременно - вж. за пример управление на LCD дисплеи в миналото занятие).

UART iii

- Какво представлява скоростта на интерфейса? Обичайно говорим или за bitrate (скорост на предаване на бит), или за baudrate (скорост за предаване на символ). Разликата между двете се определя от факта, че при е възможно един символ да не бъде бит, ами няколко бита например. Тогава, при X bitrate и символ, определен от 4 символа, baudrate ще бъде $\frac{X}{4}$. Така тези числа посочени по-рано (9600 и 115200) са всъщност baudrate.

UART iv

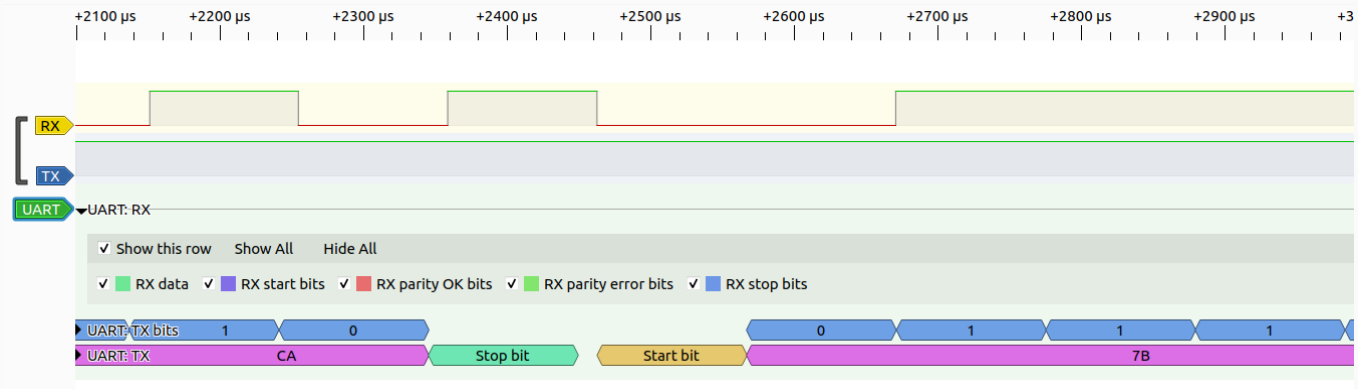
Какво е отношението между bitrate и baudrate при UART?

- Натъртваме отново - тъй като няма обособен тактов сигнал, е необходимо комуникиращите да имат предварително фиксирани скорост и формат на пакетите.
- Пренос на данни започва при настъпване на *условие на начало*.
- Пакетите имат следните основни компоненти:
  - Условие за начало,
  - Данни,
  - Проверка по четност,
  - Условие за край.

UART v

- Тъй като условието за край може да бъде един или два бита, проверката по четност може да използва различна стратегия и броя битове за данни може да варира, се използва следната нотация при договаряне между устройства: напр. 8N1 - 8 бита данни, No parity, 1 бит условие за край. Данните са в интервала 5 до 9, а вариантите за проверката по четност включват None, Even, Odd, Space, Mark.

UART vi



UART, през очите на логически анализатор.

Задача

Нека закачим два контролера Arduino Uno чрез UART и Serial библиотеката. Задачата ще бъде да реализирате играта морски шах, при която действията на всеки от играчите се предават по UART към другия участник.

I2C i

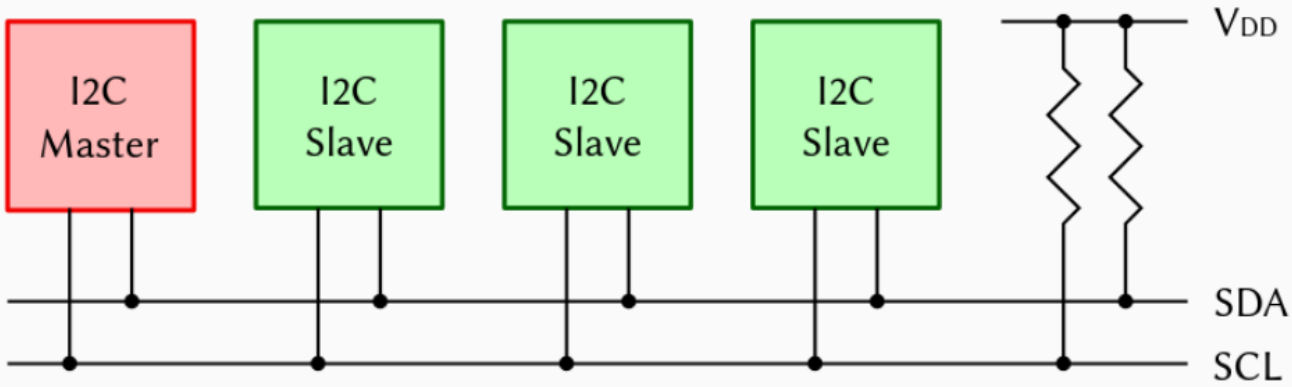
- Inter Integrated Circuit
- Създаден през 1982 г. от Phillis; с отворена (open source) спецификация от 2006 г.
- Използва се предимно за комуникация между близки (на дистанция) устройства.
- Синхронен, сериен, half-duplex, master-slave.
- Подобно на UART, използва само два проводника, но те имат доста различно предназначние при I2C. По спецификация се наричат SDA (serial data) и SCL (serial clock).

I2C ii

- Наличието на SCL сигнала премахва изискването за предварителна уговорка за *много неща*, както беше при UART.
- Скоростта на пренос на данни при е относително бавна, но не е фиксирана и може да варира между 100kb/s (Standard mode) и 1Mb/s (Fast mode plus) без промени.

Примерна I2C мрежа е посочена на Фиг. 11.

I2C iii



Фигура 11. Примерна I2C мрежа.

- Както вече отбелязахме, всяко устройство, което участва в I2C комункация е или в ролята на **master**, или **slave** (още наричани респективно adapter или bus, както и client или device).

I2C iv

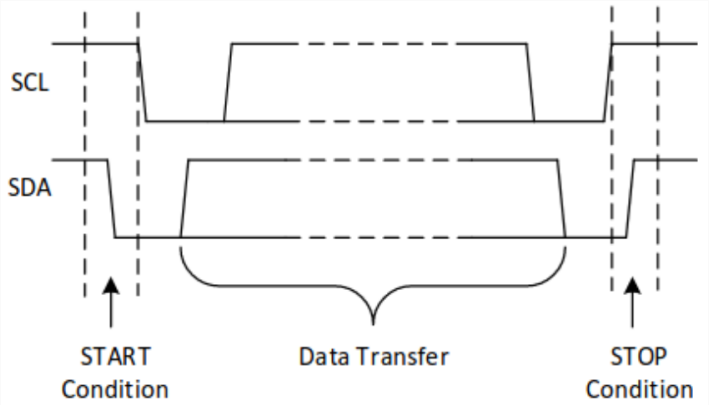
- Всяко предаване на съобщения започва от **master** устройство, когато то предаде т.нар. условие за начало.
- След началното условие е небходимо **master** устройството да избере към кое от подчинените устройства желае да се обърне, посочвайки уникален адрес (обичайно 7-битов), както и команда, бележеща дали съобщението е за писане или четене.
- Всяко съобщение е от произвола дължина - единствените задължителни компоненти са начално условие и такова за край, както и ACK бит след всеки 8 предадени бита.

I2C v

- Важно е да се отбележи, че I2C използва т.нар open-drain връзка (наместо push-pull) и поради тази причина *състоянието на всяка от шините по подразбиране е високо*.
- Броят устройства в I2C мрежа при обичайната 7-битова адресация е ограничен до 112, поради факта, че някои адреси са резервирани.
- Срещана техника е т.нар. *clock stretching*, при която slave устройството държи свалена `SCL` шината, сигнализирайки по този начин на master, че трябва да „намали темпото". Това е необходимо, тъй като именно master устройството задава тактовата честота.

I2C vi

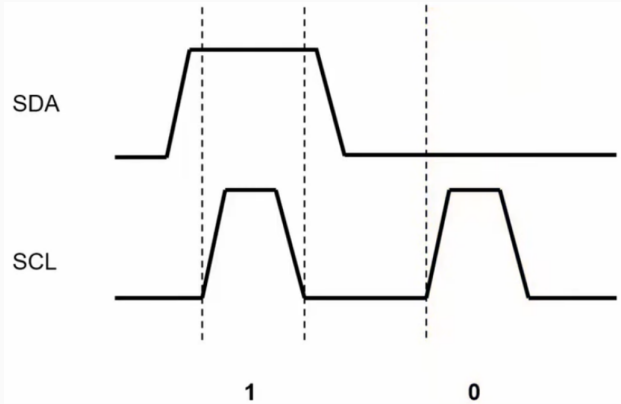
Начало на предаване на съобщение започва със START бит, а край се бележи чрез STOP бит (фиг. 11) - *само тогава меним състояние на SDA при вдигнат SCL.*



Фигура 11. Начало и край на трансакция.

I2C vii

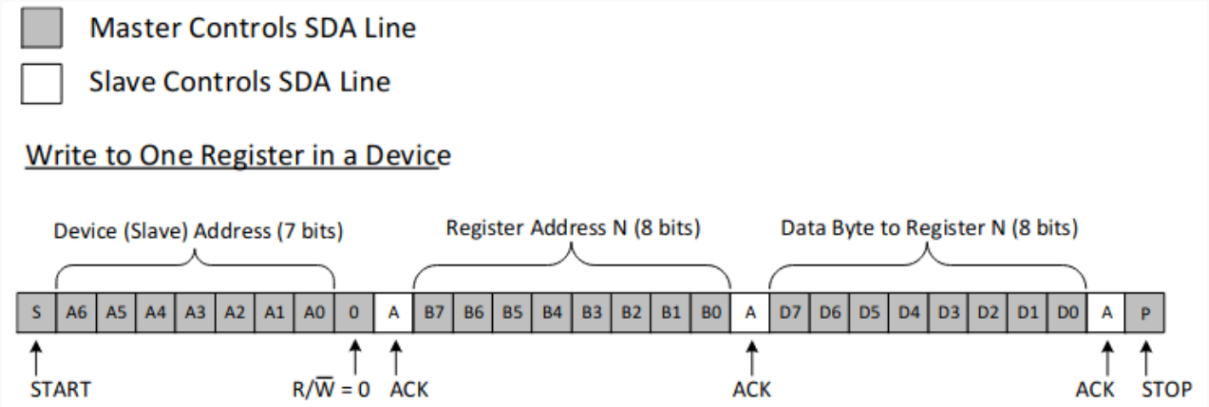
Данни предаваме като променяме състоянието на SDA, но само когато SCL и свален (фиг. 11). Те се изпращат MSB-first, а след всеки 8 се изпраща ACK бит.



Фигура 11. Пренос на данни.

I2C viii

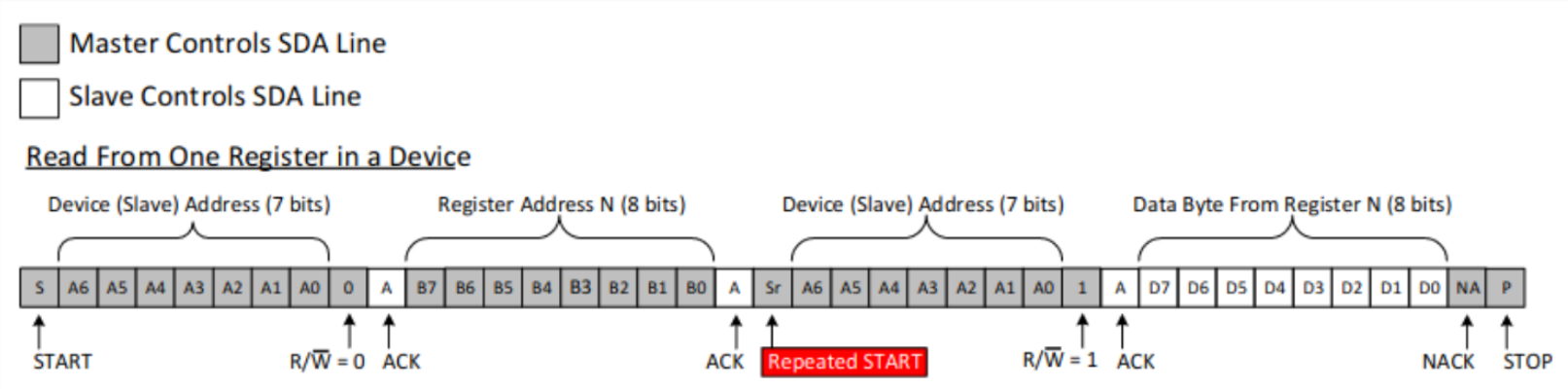
На фиг. 11 е показана примерна транскация, при която master пише върху slave.



Фигура 11. Master пише върху slave.

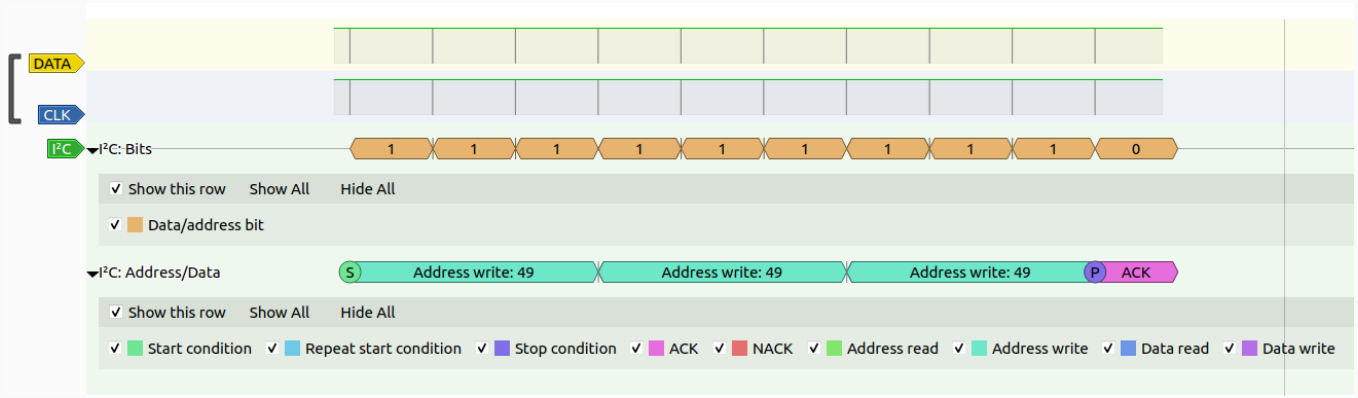
I2C ix

На фиг. 11 е показана примерна транскация, при която master чете от slave.



Фигура 11. Master чете от slave.

I2C x

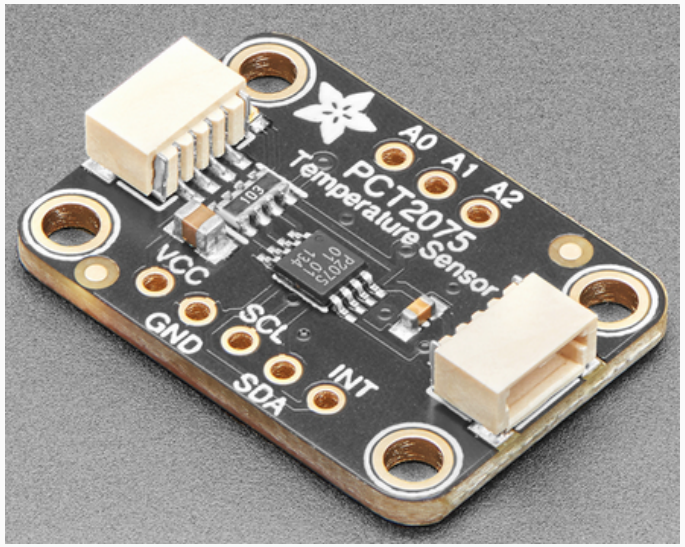


I2C, през очите на логически анализатор.

I2C xi

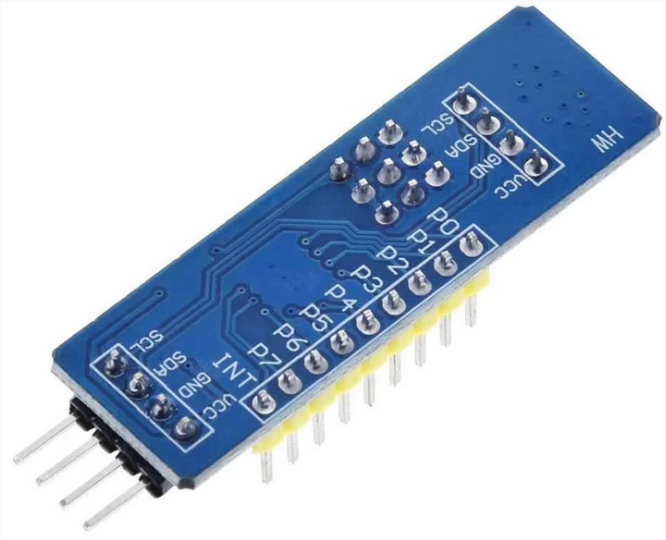
Пример

Да разгледаме документацията на сензор PCT2075.



I2C xii

Да разгледаме PCF8574.



I2C xiii

Разполагаме с библиотеката `Wire`, чрез която да използваме I2C интерфейса „наготово”. Основните функции, които ни трябват са следните:

```
Wire::begin(slave address)
Wire::onRequest(handler)
Wire::onReceive(handler)
Wire::read()
Wire::write()
Wire::requestFrom(slave address, byte cnt)
Wire::beginTransmission(slave address)
Wire::endTransmission(slave address)
```

Задача

Реализирайте следните функции, които изпращат произволен брой байтове от master към slave устройство:

```
size_t i2c_read(uint8_t addr, void *data, size_t size);
size_t i2c_write(uint8_t addr, void *data, size_t size);
```

Задача

Да променим предходната задача, така че данните да се пренасят чрез I2C, вместо чрез UART.

Задача

Свържете LCD дисплей, използващ I2C, към Arduino и реализирайте функцията `disp_write(const char *str);`, която да визуализира произволен низ на дисплея.

SPI i

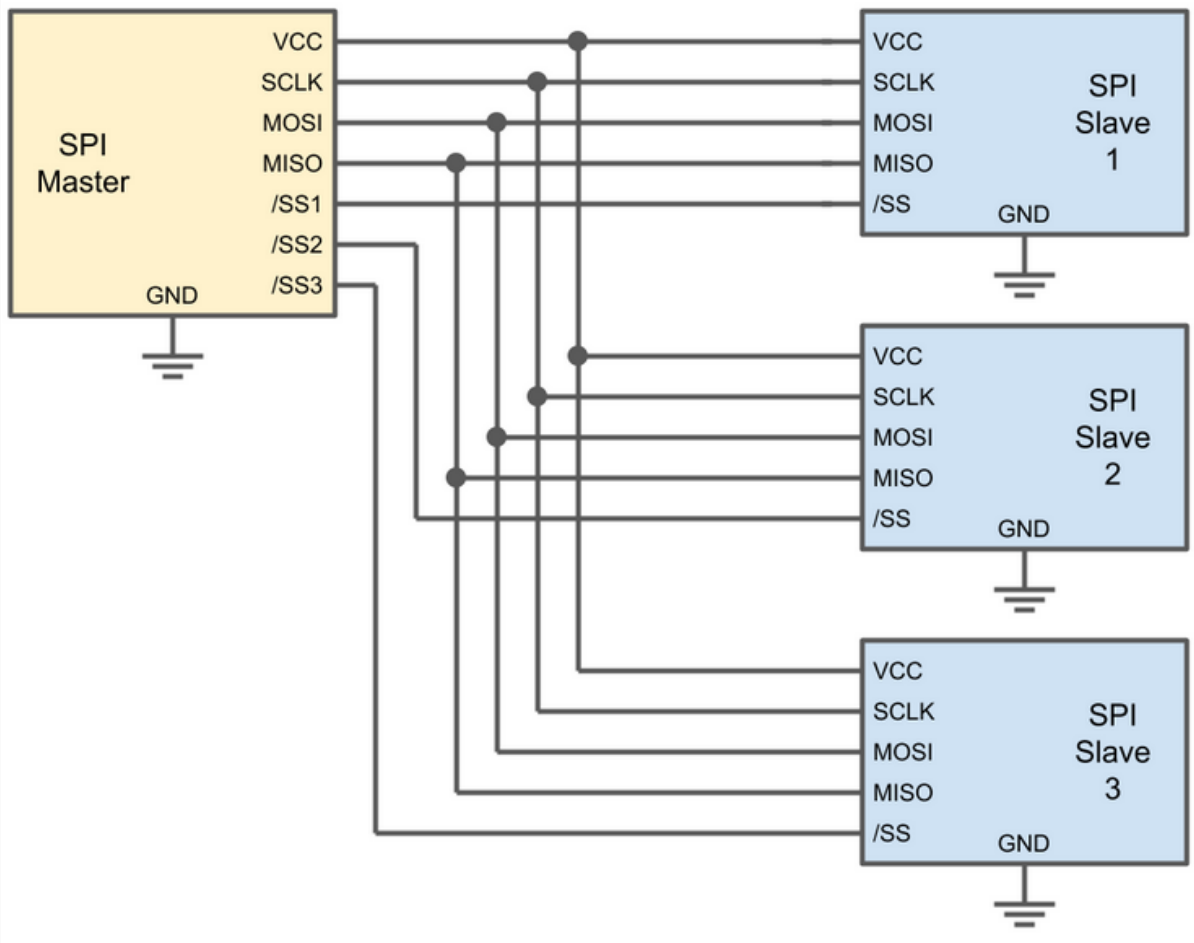
- Serial Peripheral Interface
- Motorola започват да го използват през 80-те години. За разлика обаче от I2C, SPI *няма* официално приета спецификация, която следва.
- Синхронен, сериен, full-duplex, master-slave.
- Подобно на I2C, всяко устройство, което комуникира, е или master, или slave. Разлика обаче е, че при SPI имаме единствен master, независимо от броя на периферите.
- Използва четири сигнала: `SCLK`, `MOSI`, `MISO`, `CS`. Забележете, че поради липсата на стандартизация, имената на сигналите могат да варират (напр. `DIN` вместо `MISO` или `DOUT` вместо `MOSI`).

SPI ii

- Потенциално някои периферии, които със сигурност няма да пишат по master, могат да изпуснат `MISO` (напр. LCD дисплей).

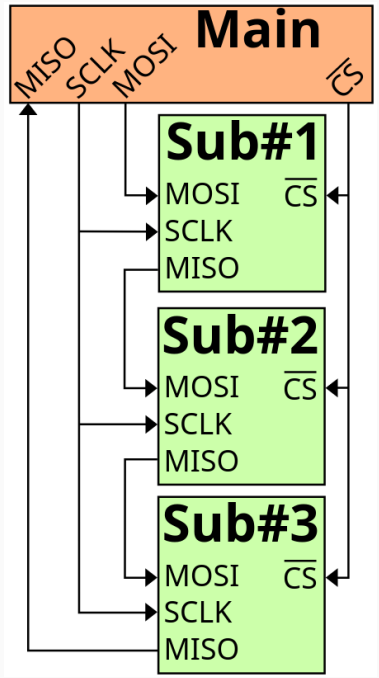
SPI iii

Основната постановка е следната:



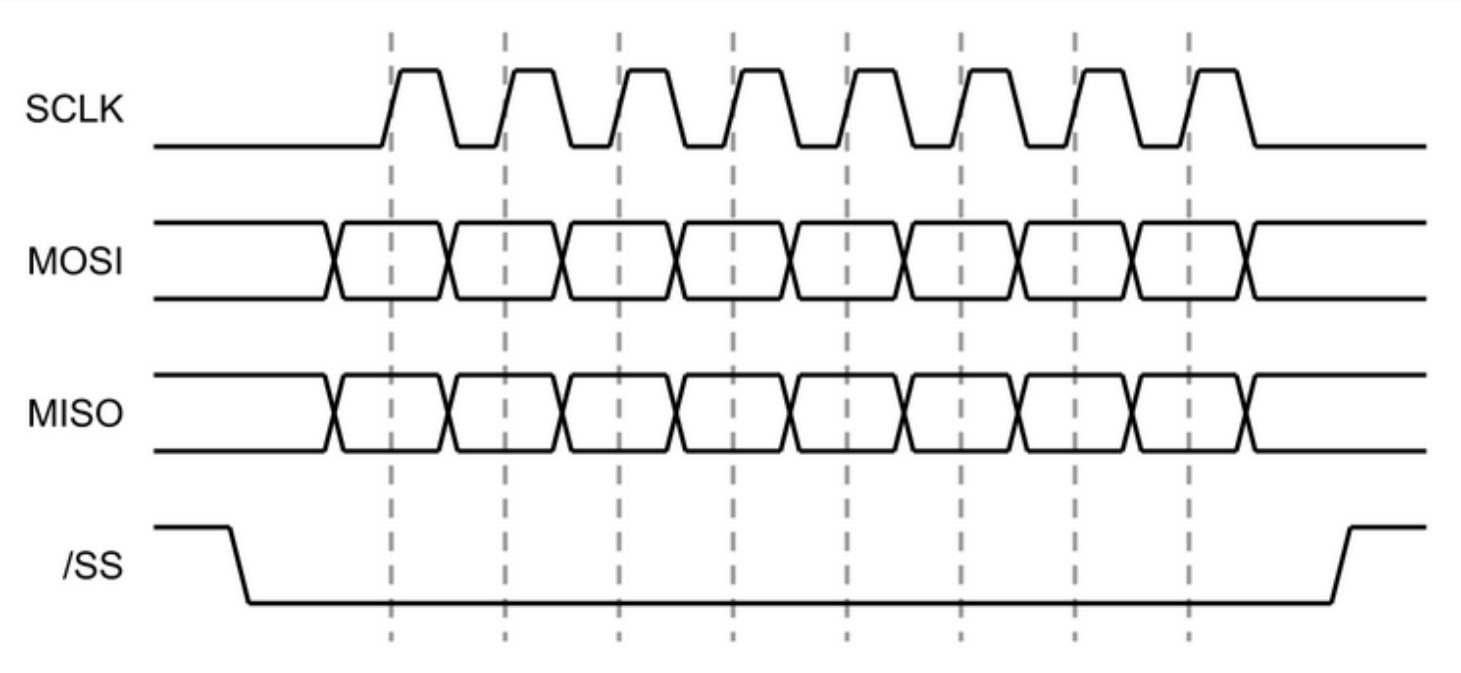
SPI iv

Друг възможен вариант е т.нар. Daisy Chaining. Забележете подобието с shift регистър.



SPI v

За да изпрати данни към периферия, първата стъпка на master устройството е да го *избере*, като свали `CS` шината. От този момент нататък при всеки такт (определен от `SCLK`) master-ът пише по `MOSI` и едновременно с това прочита от `MISO`.



SPI vi

Забележка: Както не са задължителни тези имена на сигналите, така и не е непременно вярно, че стойността на бита се извлича на преден фронт на тактовия сигнал (както е показано на фигурата).

Съществуват общо четири „*режима*" на семплиране на данните от входа и изхода на master устройството (според състояние по подразбиране на SCLK и според фронта):

SCLK по подразбиране	Фронт	Режим
0	Преден	0
0	Заден	1
1	Заден	2
1	Преден	3

SPI vii

SPI **не** уточнява скорост на предаване или метод на адресация, както прави например I2C. Поради тази причина, е важно да се запознаем с конкретната периферия, с която комуникираме преди да подходим към употребата ѝ.

Забележете, че SPI не използва open-drain изходи, ами т.нар. push-pull. Това означава, че вместо да разчитаме на резистор да ни „издърпа" до някое състояние (това по подразбиране), използваме два отделни транзистора, които да го менят.

Основните предимства на SPI пред I2C са простатата и скоростта на предаване.

SPI viii

Разполагаме с библиотеката SPI, чрез която да използваме SPI интерфейса „наготово”. Основните функции, които ни трябват са следните:

```
SPI::begin(), SPI::end()
Wire::beginTransaction(SPISettings)
Wire::endTransaction()
miso value = Wire::transfer(mosi value)
```

Забележете, че конролирането на CS е наша отговорност и не се извършва от никоя от посочените функции.

SPI ix

Завършваме темата за `I2C` и `SPI` със следната таблица за сравнение:

	SPI	I2C
Pin drive	Push-pull	Open drain
Signal lines	4 (plus 1 for each additional peripheral)	2
Max speed	No limit (10-100 Mbps is common)	400 kbps in fast mode (3.4 Mbps is possible with high-speed mode)
No. of peripherals	Only limited by number of pins available for SS lines on master	112 with 7-bit addressing
Multi-master	No	Yes
Flow control	No	Yes

Литература

- *"I2C Specification"*. URL: https://www.nxp.com/docs/en/user-guide/UM10204.pdf (дата на посещ. 17.02.2024)
- *PCT2075 Datasheet*. URL: https://gitlab.com/tues-embedded/vmks/-/blob/master/Datasheets/PCT2075.pdf?ref_type=heads (дата на посещ. 18.02.2024)
- *PCF8574 Remote 8-Bit I/O Expander for I2C Bus*. URL: https://www.ti.com/lit/ds/symlink/pcf8574.pdf?ts=1708238418992 (дата на посещ. 18.02.2024)

- *Understanding the I2C Bus*. URL: https://gitlab.com/tues-embedded/vmks/-/blob/master/Datasheets/TI_I2C_app_note.pdf?ref_type=heads (дата на посещ. 18.02.2024)
- *Interfacing LCD with I2C*. URL: https://lastminuteengineers.com/i2c-lcd-arduino-tutorial/?utm_content=cmp-true (дата на посещ. 18.02.2024)
- *I2C Bus*. URL: https://www.i2c-bus.org/ (дата на посещ. 24.02.2024)